Internet Engineering Task Force
INTERNET DRAFT
Expires December 1999

C-Y Lee
L. Andersson
Nortel Networks
Y. Ohba
Toshiba

June 1999

Avoiding Loops in MPLS
<draft-leecy-mpls-loop-avoidance-00.txt>

Status of this memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet- Drafts as reference material or to cite them other than as "work in progress."

To view the list Internet-Draft Shadow Directories, see http://www.ietf.org/shadow.html.

Abstract

This document proposes a general method to avoid loops while setting up Label Switched Paths (LSPs), and is applicable to both unicast and multicast label setup. The approach taken is to solve the problem of constructing a (loopless) tree for delivering data. The solution verifies the path towards the root of the tree is loop free before a node is grafted to the tree. This loop avoidance scheme complements loop detection in LDP.

1.0 Scope

The method described here will have its main application in ATM-LSR and FR-LSR networks or where multicast label switching is supported, but would work for any LSP setup.

The loop avoidance mechanisms are described with respect to LDP, but is applicable to other protocols that wish to setup loop-free tree as well (e.g. RSVP).

2.0 Motivation

There is no explicit loop avoidance mechanism in the current LDP. Certain scenarios avoid loops implicitly but current procedures in LDP use loop detection procedures to eliminate LSPs which are looping. However, data may already be looping before the looping LSPs are detected.

Loops are always undesirable, and more so on a tree (cf a point to point delivery path). Nevertheless, the method used to avoid loops must be such that it does not introduce a large amount of protocol overhead.

In particular, multicast loops can be harmful since packets are replicated and in the event of loops, multiple copies are generated at each loop. Multicast routing loops can affect a larger number of nodes in a network in a shorter period of time and need to be detected (and ideally prevented) before potential long lasting damage occurs.

3.0 Terminology

In MPLS, if data is flowing from node Ru to node Rd, Ru is the upstream node and Rd is the downstream node. Labels are mapped or assigned by downstream nodes and the label bindings are distributed in the "downstream to upstream" direction by means of a label distribution protocol. In an MPLS network the router are called Label Switching Routers (LSR).

An upstream router Ru may request (using a label request message) a label that it should use when forwarding packets with certain characteristics. (Forwarding Equivalence Class, FEC). A downstream router Rd would then inform (via a label mapping message) the upstream router Ru the label it should use. Rd may also distribute such a label to Ru without any prompting (i.e. label request) from Ru.

3.1 MPLS Trees

In an MPLS multipoint to point (mp2p) tree as shown in Figure 1, R4 is the downstream router Rd and R6 is the upstream router Ru.

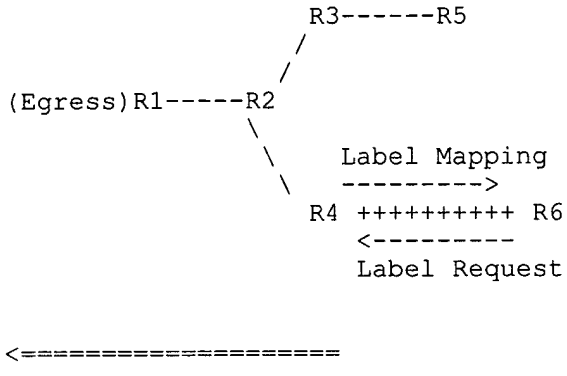


Figure 1

Figure 2 depicts an MPLS point to multipoint (p2mp) tree. In this case, R4 is the upstream router and R6 is the downstream router.

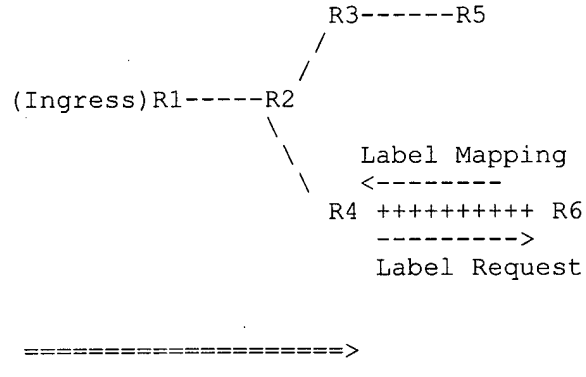


Figure 2

Note: When unicast flows are aggregated towards an egress LSR, an MPLS multipoint to point tree is setup. An MPLS point to multipoint tree is applicable to uni-directional multicast distribution trees. In the case of bi-directional shared (multicast) trees, data flows towards R1 as well as away from R1. However the same loop avoidance scheme can be used for both uni-directional and bi-directional trees.

See section 3.1.1.

3.1.1. Multicast MPLS Trees

Multicast trees are categorized in three types: source tree, unidirectional shared tree, and bidirectional shared tree.

In the case of source tree, the root of the MPLS tree is the ingress LSR of the source tree.

In the case of unidirectional shared tree, the root of the MPLS tree is either the core node (e.g., the Randevouz Point of PIM-SM), or the ingress LSR of the MPLS tree (if the core node is not included in the MPLS tree).

In the case of bidirectional shared tree (i.e., CBT), the root of the MPLS tree is either the core node (i.e., CBT core node), or the LSR which is nearest to the core node among LSRs on the MPLS tree (if the core node is not included in the MPLS tree).

4.0 Avoiding loops in LSP

Currently, an LSR is attached to an MPLS tree when it first receives an outgoing label mapping from an acceptable downstream neighbor.

The fundamental problem to solve here is how to attach a node to a tree without causing loops in the resulting tree.

We have identified two distinct cases to consider: i) when attaching a single node to a tree ii) when attaching a sub-tree (labeled path) to a tree (labeled path)

The loop avoidance scheme here requires the LSR to be attached to an MPLS tree, to verify the path towards the root of the tree is loop free when one labeled path is spliced with another labeled path.

4.1 Procedure

The basic idea used here to avoid loop in a tree is independent of

the direction of data flow and the type of MPLS tree. However the loop avoidance scheme will be described in terms of the label setup procedure and hence the role of the downstream and upstream LSR in the scheme is reversed for each type of MPLS tree (as implied by Figure 1 and Figure 2).

The two distinct cases listed in the section "Avoiding loops in LSP", correspond to these two conditions in MPLS, respectively:

1) when an LSR receives (mp2p tree) or send (p2mp tree) a label mapping which it has no binding yet. This event may be preceded by Ru sending a label request to Rd. Eg. when Ru initially attempts to setup an LSP in the egress direction.

2) when an LSR receives (mp2p tree) or send (p2mp tree) a label mapping for a label which it already has binding. This event may also be preceded by Ru sending a label request to Rd. Eg when the FEC (associated with this label) next hop changes.

Upon receiving (mp2p tree) or before sending (p2mp tree) a label mapping, an LSR Rx would verify which of the above conditions is true:

a) If case (1), then the label mapping is accepted (mp2mp) or sent (p2mp) and no additional new actions are needed.

b) If case (2), Rx will send a 'special' label request message, label splice message. This message must be forwarded towards the root of the MPLS tree (egress LSR for mp2p and ingress for p2mp) ie along the already labeled path. The last LSR on the already established LSP path will send a label splice acknowledgement message back towards the same path where the label splice message was sent. Once Rx received the acknowledgment message, the label mapping is accepted (mp2p tree) or sent (p2mp tree). In other words, the sub-tree will be spliced with the tree.

If an LSR receives a label splice message and it already has a pending splice message, the LSR knows there is a possibility of loop and takes an appropriate action so that Rx will not receive the acknowledgment message if there is a loop, thereby preventing a looping LSP from being established.

We are currently investigating two choices for the appropriate action. One choice is to merge the receiving label splice message with the pending splice message. The other is not to merge the receiving message.

Note that Rx is the upstream router in the MPLS mp2p tree case. In

the MPLS p2mp tree case, Rx is the downstream router. Essentially, the node (R6 in Figure 1 and Figure 2) that is going to be attached to the tree will send the label splice message.

How does an LSR know if it is attaching to a mp2p tree or p2mp tree? The LSR can infer from the FEC (unicast or multicast address) the type of tree it will be attaching to.

4.2 Non-merging LSRs

The above procedures are only necessary for merging LSRs since labeled paths are never merged (spliced) by non-merging LSRs. Hence non-merging LSRs does not cause data to loop. A different label mapping is returned for each label requested (i.e. the labels are not merged). Note that label mappings are not distributed to non-merging LSRs unless requested (label request message).

4.3 Looping control messages

A merging LSR will not forward label request messages if there is already a pending label request for that FEC, but instead will attempt to merge the request once it receives the corresponding label mapping (in this case it will not receive the label mapping since the request message is looping). Hence a merging LSR will not cause a label request message to loop.

A non-merging LSR, however does not merge the label request message. It will provide a different label mapping for every label request message it receives and forward the message to the egress router. Hence if there is a routing loop, a request message may loop indefinitely.

We do not view looping control messages as serious a threat as looping data packets. LDP [LDP] has procedures to detect this kind of loop and they should be adequate to deal with looping control messages.

4.4 Ordered vs. Independent Control mode

The MPLS architecture allows labels to be used for data before the LSPs have been completely setup. Ideally labels should be used for multicast data forwarding only after the branch of an LSP have been completely setup to reduce the effects of incorrectly labeled packets from being multicasted in a network.

Note that the Label Splice mechanisms, however, are orthogonal to whether LSRs are using independent control mode (where labels can be used for data before the LSPs have been completely setup) or ordered

control mode (where a label is not distributed and used until an LSR receives the label mapping/binding for the corresponding FEC from its next hop for the FEC).

4.5 Why this scheme is simple?

The proposed scheme is simpler than the colored thread algorithm [LOOP]. This is due to : (i) separation of functionality of LSP loop prevention and functionality of prevention of label request message looping, (ii) separation of label mapping phase and label splicing phase, and (iii) the fact that the label splicing message itself has no special information for loop prevention other than FEC and label.

At the cost of (i), the scheme itself does not have a functionality of prevention of label request message looping. However, the LDP already has a method to prevent label request message looping and this will not become a problem.

At the cost of (ii), in some cases more messages would be required than the colored thread algorithm when a route changes. However, the cost would be acceptable, since the label splice messages are always forwarded toward the root of the tree regardless of whether the tree is p2mp or mp2p and can be merged at each branch of the MPLS tree.

At the cost of (iii), the scheme does not have a functionality to explicitly "detecting" LSP loop. However, this will not become a problem, because the main objective of loop prevention is not to detect an LSP loop but to prevent an LSP from forming a loop. Note: if this scheme is adopted in LDP, it should be used together with the LDP loop detection, and the loop detection will detect LSP loops (see section 5.0).

5.0 Interoperability

In the case of LDP, there would be a case in which some LSR is performing the proposed loop prevention scheme while other LSR is performing loop detection based on path vector.

Suppose that Ru and Rd are LDP peers, and Ru and Rd are performing the loop prevention and loop detection, respectively. Ru never sends a label splicing message to Ru. On the other hand, Ru may receive a label mapping message with a path vector but without a hop count from Rd. If Ru does not forward the label mapping message including a path vector to upstream LSRs, there is a possibility of forming an LSP loop since the information needed for loop detection is completely lost.

In order to avoid this kind of interoperability problem, an LSR which performs the proposed loop avoidance scheme must also performs the procedure required for the LDP loop detection when it receives a label request or label mapping message containing a path vector. Hence when the P-bit is set, the D-bit is set too. See Packet Format section.

I.e, when an LSR receives a label request message with a path vector, it adds its own address to the path vector and forward the label request message with the path vector to the downstream LSR, unless label request message looping is detected.

On the other hand, when an LSR receives a label mapping message with a path vector, it adds its own address to the path vector and forward the label mapping message with the path vector to each of the upstream LSRs, unless LSP loop is detected. The LSR may also originate a label splicing message as a result of receiving/sending the label mapping message. In this case, label switching between incoming and outgoing labels is kept pending until it receives an ACK for the label splice message.

If an LSR Rd that is performing the proposed loop avoidance scheme receives a label splice message from Ru and the next hop LSR to the root of the MPLS tree is not performing the proposed loop avoidance scheme, Rd should immediately return an ACK to Ru instead of forwarding the label splice message further.

5.0 Packet Format

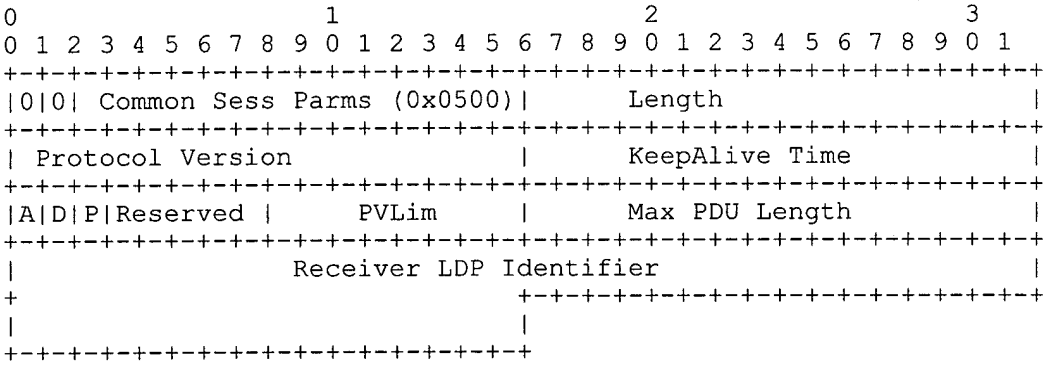
The new LDP TLVs (Type, Length, Value) [See LDP Specification] required are:

- Label Splice Message This will contain the Label Splice Message type, the message length, message ID, the address of the LSR which originates the splice message, FEC TLV and Label TLV.

- Label Splice Acknowledgment Message This will contain the Label Splice Acknowledgment Message type, the message length, message id, the address of the LSR which originates the splice message, FEC TLV and Label TLV.

5.1. Changes in LDP Common Session Parameters TLV

A new one-bit field is defined in Common Session Parameters TLV.



P, Loop Prevention

Indicates whether loop prevention based on the proposed scheme is enabled. A value of 0 means loop prevention is disabled; a value of 1 means that loop prevention is enabled. When P-bit is set to 1, D-bit must also be set to 1 (see section 5.1).

No label splice message is sent to an LDP peer from which a Common Session Parameters TLV is received with P-bit=0.

6.0 Acknowledgments

The authors would like to thank Joel Halpern and Peter Ashwood-Smith for their helpful comments.

References

[ARCH] E. Rosen, A. Viswanathan, R. Callon, "Multiprotocol Label

Switching Architecture", Work in Progress, July 1998.

[ATM] B. Davie, J. Lawrence, K. McCloghrie, Y. Rekhter, E. Rosen, G. Swallow, P. Doolan, "Use of Label Switching With ATM", Work in Progress, September, 1998.

[CRLDP] L. Andersson, A. Fredette, B. Jamoussi, R. Callon, P. Doolan, N. Feldman, E. Gray, J. Halpern, J. Heinanen T. E. Kilty, A. G. Malis, M. Girish, K. Sundell, P. Vaananen, T. Worster, L. Wu, R. Dantu, "Constraint-Based LSP Setup using LDP", Work in Progress, January, 1999.

[ENCAP] E. Rosen, Y. Rekhter, D. Tappan, D. Farinacci, G. Fedorkow, T. Li, A. Conta, "MPLS Label Stack Encoding", Work in Progress, July,

1998.

[FR] A. Conta, P. Doolan, A. Malis, "Use of Label Switching on Frame Relay Networks", Work in Progress, October, 1998.

[LOOP] Y. Ohba, Y. Katsube, E.Rosen, P.Doolan, "MPLS Loop Prevention Mechanism", Work in Progress, May 1999.

Authors' Information

Cheng-Yin Lee
Nortel Networks
PO Box 3511, Station C
Ottawa, ON K1Y 4H7, Canada
leecy@nortel.com

Loa Andersson
Nortel Networks Inc
Kungsgatan 34, PO Box 1788
111 97 Stockholm
Sweden
Phone: +46 8 441 78 34
Mobile: +46 70 522 78 34
email: loa_andersson@baynetworks.com

Yoshihiro Ohba
Toshiba Corporation
1, Komukai-Toshiba-cho, Saiwai-ku
Kawasaki 210-8582, Japan
email: yoshihiro.ohba@toshiba.co.jp